CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

A Rough-Cut Capacity Planning Model with Overlapping

Georges Baydoun
Alain Haït
Robert Pellerin
Bernard Clément
Guillaume Bouvignies

December 2014

CIRRELT-2014-65

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · McGill · Concordia · ÉTS · UQÀM · HEC Montréal · Polytechnique Montréal · Université de Montréal

A Rough-Cut Capacity Planning Model with Overlapping

Georges Baydoun[1,2], Alain Haït[3], Robert Pellerin[1,2], Bernard Clément[2], Guillaume Bouvignies[3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematical and Industrial Engineering, Polytechnique Montréal, C.P. 6079, succursale Centre-ville, Montréal, Canada H3C 3A7

[3] University of Toulouse, Institut Supérieur de l'Aéronautique et de l'Espace, Supaero, 10, avenue Édouard-Belin, B.P. 54032, 31055 Toulouse Cedex 4,Toulouse, France

Abstract. In this article, we propose an event-based mixed integer linear programming model for the Rough-Cut Capacity Planning (RCCP) problem with different feasible overlapping modes between work packages. In the model, time horizon is divided into time buckets used to evaluate resource usage, while starting and ending times for work packages are continuous. The model was tested on a benchmark of 5 sets of 450 instances each. More than half of tested instances were solved to optimality within 500 seconds. Results also prove that, while overlapping is more beneficial for accelerating project delivery times, it can still have a positive impact on project cost by allowing a better distribution of workload. Finally, overlapping options seem to have less influence on the performance of the model than project slack or number of work packages.

Keywords. Rough-cut capacity planning, concurrent engineering, overlapping.

Acknowledgments. This work has been supported by the Natural Sciences and Engineering Research Council of Canada and the Jarislowsky/SNC-Lavalin Research Chair in the Management of International Projects. Their support is gratefully acknowledged.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: Robert.Pellerin@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2014

© Baydoun, Haït, Pellerin, Clément, Bouvignies and CIRRELT, 2014

1 Introduction

Overlapping is a common practice used in construction and in product development projects to accelerate the execution of large projects. This technique consists in executing in parallel two sequential activities by allowing a downstream activity to start before the end of an upstream activity based on preliminary information. However, the overlapping of activities can entail rework tasks and modifications further to the transmission of complementary information after the start of the downstream activity (Berthaut et al., 2014). Activity overlapping thus allows reducing the total duration of project execution, at the expense of additional workload and execution cost associated to rework tasks.

Several authors have studied the relation between rework and the amount of overlap in project conducted in a concurrent engineering context or fast-tracking mode. For instance, Gerk and Qassim (2008) proposed a linear model for project acceleration using crashing, overlapping and activity substitution. Activity crashing includes the allocation of additional resources for an activity in order to accelerate its execution. Activity substitution is another technique for accelerating projects by replacing one (or more) activity by another activity.

Berthaut et al. (2014) and Grèze et al. (2012) proposed a more realistic approach by restricting overlapping possibilities to a set of feasible overlap durations for each couple of overlappable activities, instead of considering a continuous and linear relation between overlap amount and rework. This assumption is more realistic as overlapping points between activities are defined through clear document or information exchange in a concurrent engineering context, which limits the overlapping modes to a reduced and discrete set of possibilities.

However, these Resource-Constrained Project scheduling problem (RCPSP) models are not suited for planners in the early phases of projects as detailed activity content, durations, and resources are not known with precision and as work intensity cannot be assumed to be constant over execution time. Indeed, planners tend to adopt an aggregate planning approach in large engineering projects where Work Packages (WPs) are broadly defined as group of multiple activities that could extend on a long period, i.e. weeks or months (Cherkaoui et al., 2013). In practice, project planning is done by preparing several schedules at different phases of the project, where aggregation levels depend on the ongoing phase and on the targeted audience. For instance, the Front-End-Loading (FEL) approach, commonly used in large construction projects, is composed of successive planning stages. Early phases involve tactical planning based on Rough-Cut Capacity Planning (RCCP) techniques in order to fix all project milestones and estimate resource usage. At this level, projects are divided into work packages (WPs) which are clusters of activities. Rough-Cut Capacity Planning (RCCP)

models divide the planning horizon into time buckets (or periods) used to evaluate critical resource usage and by allowing resource allocation to WP to vary from one period to another De Boer (1998).

Recognizing the need of practitioners to better support the project planning function in the early phases of projects, this paper proposes an exact RCCP model that determines the order of execution in time of a set of WPs so as to minimize the total project duration and/or project cost, while respecting precedence relations, resource constraints and considering overlapping possibilities. The proposed model considers variable WP intensities and aggregate resource capacities.

The reminder of the paper is organized as follow. We first give a brief state of the art of existing RCCP models and overlapping models in section 2. We then introduce the original mixed-time RCCP model in section 3, before explaining our new RCCP model with multiple overlapping modes in section 4. Section 5 explains the generation of our test instances, and an illustrating example is presented in section 6. Finally, section 7 analyzes the performance and the results of our model, before giving some concluding remarks in section 8.

2 Related work

In the order acceptance stage of a project, companies tend to commit to due dates without an accurate knowledge of their resource capacities. The Rough-Cut Capacity Planning (RCCP) ensures, at an aggregated level, that the capacities of critical resources are sufficient to complete a project within its time and cost limits (De Boer, 1998). Performed at the tactical level, the RCCP is based on a horizon divided into time buckets (or periods) used to evaluate the resource usage. The WPs are defined by their work content and resource allocation can vary from one period to another. Capacity and resource allocation flexibility allow to adapt the WP durations according to time and cost-related considerations. WPs may start or end during a period hence it is possible to plan a WP and its successor within the same period.

Besides denominated RCCP models, RCPSP models where intensities can vary from period to period are also suitable for the RCCP problem as they consider fixed workload for each activity and variable resource usage between periods, and therefore variable activity durations. These models have several appelations in the literature such as Resource Constrained Project Scheduling with Variable Intensity Activities (RCPSVP) and RCPSP with flexible resource profiles.

Wullink (2005) distinguishes three classes of solution approaches for the RCCP : straightforward constructive heuristics, LP based heuristics and exact approaches. Among existing

RCCP models, Hans (2001) proposed an exact approach that consists in determining the periods where each job can be executed, and then specifying the fractions of the WP contents that are actually executed in each period. However, this method can allow a predecessor and any direct successor to be performed in the same period without determining their ending and starting times within the period, which could lead to precedence infeasibility. Hans (2001) proposed two alternatives to avoid this problem by over-constraining the problem. Taking a project scheduling point of view, Kis (2005) proposed an RCPSP model with variable activity intensities, that forbids two activities with direct precedence relation to be executed in the same period.

More recently, Haït and Baydoun (2012) proposed an exact approach that consists of using continuous time representation for events, together with a discrete evaluation of resources. This means that start and ending dates of WPs can be determined within a time period, making it possible to guarantee the respect of precedence constraints. Yet, resource consumptions are still evaluated globally over periods, making the model suitable for planning at a tactical level where planning is performed in practice on a period-by-period basis (i.e. resource availabilities and allocations are determined per period).

Lately, Naber and Kolisch (2014) addressed the RCPSP with flexible resource profiles, meaning that resource usage of an activity can vary from period to period. They propose four different discrete-time MILP model formulations based on existing formulations and compare their performance. Their experiments show that the model called "FP-DT3", that is based on the RCPSP formulation of Bianco and Caramia (2013) and the RCPSVP model of Kis (2005), dominates the other formulations in both solution quality and run-time. Several heuristic approaches have also been proposed to solve the RCCP problem including constructive heuristics (De Boer, 1998) and linear-programming based heuristics (Gademann and Schutten, 2005).

Although some authors proposed interesting extensions to the precedence relations of RCPSVP models by allowing overlapping, none of these models considers overlapping and rework. In fact, Kis (2006) extended his RCPSVP model by introducing feeding precedence constraints : a successor can start after a percentage of the execution of his direct predecessor is completed. Alfieri et al. (2011) extended feeding precedence constraints to generalized precedence relations. However, both Kis (2006) and Alfieri et al. (2011) do not take reworks into account. Moreover, these models have only one possibility of precedence constraints and do not consider several feasible modes of overlapping with different amounts of rework.

Despite these recent advances, these models do not allow overlapping of WPs that are normally executed consecutively. This concurrent engineering method is a trend in product

development and is also widely used in contexts such as construction in order to accelerate project execution. It also adds flexibility in starting and ending times, allowing a better usage of regular capacities thus reducing the need for external resources. However overlapping adds workloads, "reworks", on both down-stream and up-stream WPs (Berthaut et al., 2014). Reworks are due to information exchange and eventual alteration of the work, that are caused by starting a down-stream WP before all finalized information is available from up-stream WP.

To fill this gap, we propose a mixed integer linear-programming model, that is an extension of the RCCP model proposed by Haït and Baydoun (2012), where predecessor-successor WPs can overlap according to multiple overlapping modes. This extended model assumes that each overlapping mode can be defined by the percentage of execution of predecessor WP that needs to be reached in order to start the successor, as well as the amount of reworks on both WPs. The model is further explained in the following section.

3 Mixed-time RCCP model

This section presents the original model of Haït and Baydoun (2012). It combines a continuous time representation of events and a discrete time evaluation of resources. A set of binary variables ensures the relation between starting time, ending time and durations over periods. These durations give minimum and maximum workload that can be assigned to the period.

In this section only, the starting and ending times of WP i are denoted by t_i^0 and t_i^1, respectively. Binary variables z_{ip}^0 and z_{ip}^1 equal 1 if the starting and ending times (respectively) of WP i occur during or before time period p. Variables d_{ip} and l_{ip} are respectively the duration and the assigned workload of WP i over the period p. Finally, $Succ_i$ represents the set of successors of i. In section 4, the aforementioned notations are generalized in order to handle overlapping. Other definitions that are relevant to this section are given in tables 0.1 and 0.2.

In addition to basic definition-related constraints, constraints for the model of Haït and Baydoun (2012) are of four types : those ensuring the link between continuous and binary variables for starting and ending times, the ones concerning the durations over periods, scheduling constraints, and finally workload and intensity constraints.

3.1 Continuous/Binary variables constraints

The following constraints (1) to (3) ensure that binary variables z_{ip}^0 and z_{ip}^1 equal 1 only in the periods during or after starting and ending events (respectively). These binary variables were first introduced by Pritsker and Watters (1968) as a step formulation for scheduling with

limited resources, and used by several other authors including Bianco and Caramia (2013). However, only the model of Haït and Baydoun (2012) used them in the context of a mixed continuous and discrete time representation.

$$t_i^c \geq D \cdot p \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0,1\}, p \in P \qquad (1)$$

$$t_i^c \leq D \cdot p + H \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0,1\}, p \in P \qquad (2)$$

$$z_{ip+1}^c \geq z_{ip}^c \qquad \forall i \in I, c \in \{0,1\}, p \in \{1..|P|-1\} \qquad (3)$$

3.2 Durations over periods

When considering a WP i and a time period p, starting and ending events can be before, during, or after p. Therefore, each couple WP i and time period p has six possible configurations, depicted in Figure 0.1.

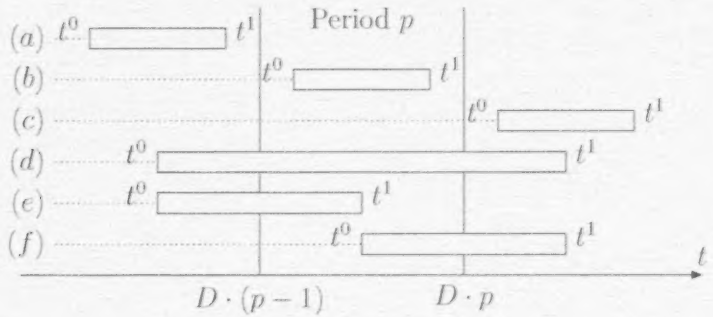

Figure 0.1 The six possible configurations for a WP regarding a time period p

The following constraints give the relations between WP durations over periods d_{ip}^1 on one side, and binary variables z_{ip}^0 and z_{ip}^1, and continuous time variables t_i^0 and t_i^1 on the other side :

$$d^1_{ip} \leq D \cdot (z^0_{ip} - z^1_{ip-1}) \qquad \forall i \in I, p \in 1..|P| \qquad (4)$$

$$d^1_{ip} \geq D \cdot (z^0_{ip-1} - z^1_{ip}) \qquad \forall i \in I, p \in 1..|P| \qquad (5)$$

$$d^1_{ip} \geq t^1_i - D \cdot p + D \cdot z^0_{ip-1} - H \cdot (1 - z^1_{ip}) \qquad \forall i \in I, p \in 1..|P| \qquad (6)$$

$$d^1_{ip} \geq D \cdot p \cdot (1 - z^0_{ip-1}) - t^0_i - D \cdot z^1_{ip} \qquad \forall i \in I, p \in 1..|P| \qquad (7)$$

$$\sum_{p=1}^{|P|} d^1_{ip} = t^1_i - t^0_i \qquad \forall i \in I \qquad (8)$$

Constraints (4) force d^1_{ip} to 0 when WP i is not active during period p (configurations (a) and (c)) and limit its value to period duration D if i and p are in any of the four other configurations. Inequalities (5), (6), and (7) concern configurations (d), (e), and (f) respectively. Inequalities (5) give d^1_{ip} lower-bounds if WP i begins before period p and is not completed during p. Constraints (6) (respectively(7)) provide lower-bounds for d^1_{ip} when WP i is finished (respectively started) in period p. Finally, constraints (8) provide global coherence between starting and ending times, and durations per periods.

3.3 Scheduling constraints

Each WP has its own time window, consisting of a release date RD_i and due date DD_i. Constraints (9) and (10) ensure that WP i is executed in its allowed time window. Moreover, a successor cannot start before its predecessor is completed (constraints (11)).

$$t^0_i \geq RD_i \qquad \forall i \in I \qquad (9)$$

$$t^1_i \leq DD_i \qquad \forall i \in I \qquad (10)$$

$$t^0_j \geq t^1_i \qquad \forall i \in I, j \in Succ_i \qquad (11)$$

3.4 Workload and intensity constraints

The workload assigned to a period should respect minimum and maximum allowed intensities BJ_i^{min} and BJ_i^{max} (constraints (12) and (13)), and the total assigned workload over time horizon should match the required workload for the WP (equalities (14)).

$$l_{ip}^1 \geq BJ_i^{min} \cdot \frac{d_{ip}^1}{D} \qquad \forall i \in I, p \in P \tag{12}$$

$$l_{ip}^1 \leq BJ_i^{max} \cdot \frac{d_{ip}^1}{D} \qquad \forall i \in I, p \in P \tag{13}$$

$$\sum_{p \in P} l_{ip}^1 = L_i^{total} \qquad \forall i \in I \tag{14}$$

3.5 Definition-related constraints

The following constraints are straight-forward results of definitions of *makespan*, regular and non regular resource allocations y_{rp}^{int} and y_{rp}^{ext}, and *cost*.

$$makespan \geq t_i^1 \qquad \forall i \in I \tag{15}$$

$$y_{rp}^{int} + y_{rp}^{ext} \geq \sum_{i \in I} A_{ir} \cdot l_{ip}^1 \qquad \forall r \in R, p \in P \tag{16}$$

$$y_{rp}^{int} \leq K_{rp}^{int} \qquad \forall r \in R, p \in P \tag{17}$$

$$cost = C_{ext} \cdot \sum_{r \in R, p \in P} y_{rp}^{ext} \tag{18}$$

Variables y_{rp}^{int} and y_{rp}^{ext} are respectively regular and non-regular allocations for resource r during period p. While we dispose of free K_{rp}^{int} regular capacity for resource r during period p (constraints (17)), non-regular capacities come at a price. Project cost is calculated as the total cost of non-regular capacities used for the whole project (equations (18)).

Possible objective functions are project makespan, project cost, or a trade-off between makespan and cost.

4 Mixed-time RCCP model with overlapping

Our new model is based on the same representation of time and events of WP start and WP end, but adds a third type of events : intermediate milestone attainment. These milestones are events that signal the attainment of a certain development state of a WP. They could correspond to important decision making moments, or the completion of key deliverables.

An overlapping mode between a WP i and its successor j is defined regarding the attainment – in terms of workload – of a milestone within WP i ; the successor can start once the milestone is reached. An overlapping mode for a WP i is a combination of different overlapping modes

between WP i and every successor of WP i that can overlap on i. Each WP i is therefore divided into $C_i + 1$ parts, where C_i is the number of successors of WP i that can overlap on it. The end of each part matches the time when a milestone of WP i is reached.

Take the example of a WP A with two successors that can overlap B and C. WP A is therefore devided into three parts ($C_A = 2$) in this example. WPs B and A have two overlapping modes : B can start after 70% or 100% (no overlapping) of A is completed in terms of workload. WPs C and A also have two overlapping modes at 35% and 100% of the total workload of A. This means that WP A has four overlapping modes with its successors as depicted in Figure 0.2. If overlapping mode 3 is selected for WP A, then variable e_{Am} equals 1 for $m = 3$ and 0 for $m \in \{1, 2, 4\}$.

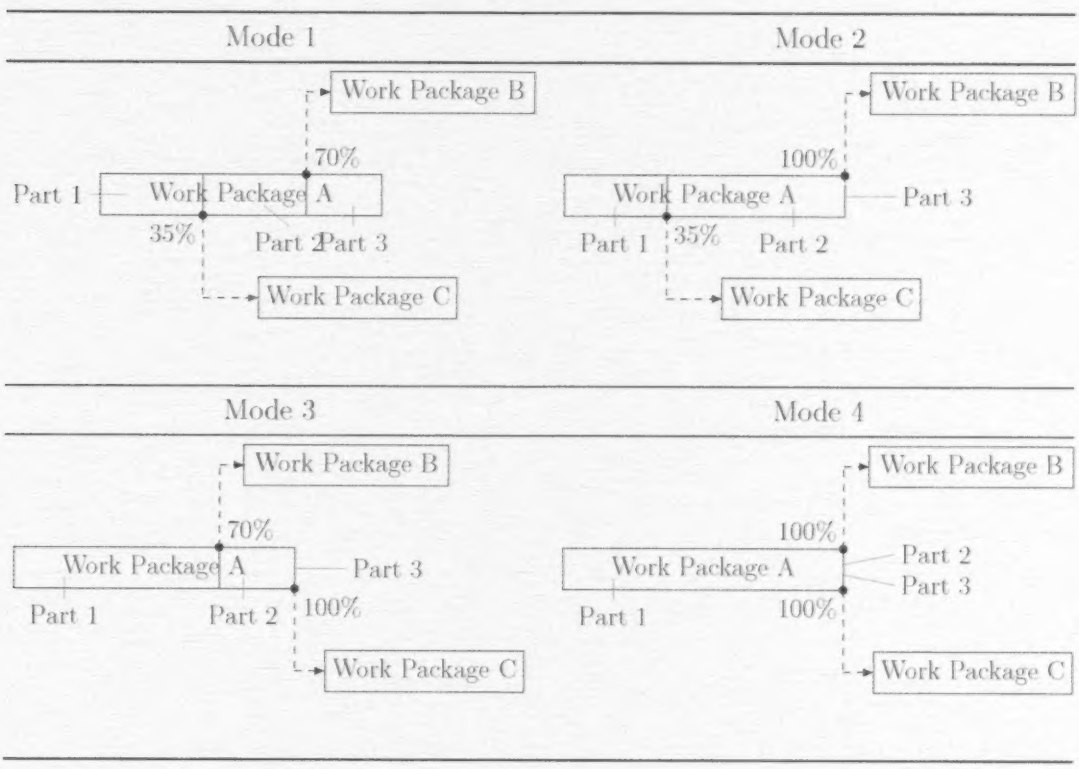

Figure 0.2 Four possible overlapping modes for a WP A with two successors B and C that can overlap on A

Each part c has its own set of durations over the periods, denoted d^c_{ip}, and assigned workload during the periods, l^c_{ip}. We also add overlapping durations over periods d_{ijp} for overlapping WPs. Variables d^c_{ip} and d_{ijp} guarantee that workloads (l^c_{ip}) and reworks ($l^{pred}_{ijp}, l^{succ}_{ijp}$) always respect the minimum and maximum allowed workload intensities. Tables 0.1 and 0.2 give the full nomenclature for our new RCCP model with overlapping.

Tableau 0.1 Nomenclature : sets & parameters

Set / Parameter	Description
P, D, H	Set of time periods ($p \in P$), duration of a period, time horizon $H = D \cdot \|P\|$
I	Set of work packages ($i \in I$)
RD_i, DD_i	Ready date (respectively due date) of i
R	Set of resources ($r \in R$)
K_{rp}^{int}	Available regular capacity of resource r during period p
A_{ir}	Percentage of consumption of resource r by WP i
BJ_i^{min}, BJ_i^{max}	Minimum and maximum workload that can be assigned for i during a duration of D
$Succ0_i$	Set of successors of i that cannot overlap on i ($j \in Succ0_i$)
$Succ1_i, C_i$	Set of successors of i that can overlap on i ($j \in Succ1_i$), $C_i = \|Succ1_i\|$
$Pred1_i$	Set of predecessors of i that can overlap on i ($i_0 \in Pred_i$)
M_i	Set of overlapping modes between i and its direct successors ($m \in M_i$)
Pos_{ijm}	Position of j among the successors of i according to mode $m \in M_i$
Ch_{ijm}	Binary parameter that equals 1 if WPs i and j overlap in mode $m \in M_i$
L_i^{total}	Total required workload for WP i
L_{im}^c	Required workload of part c of WP i in mode m
$L_{ijm}^{pred}, L_{ijm}^{succ}$	Required rework on i (respectively on j) in mode $m \in M_i$ due to overlapping between i and j
C_{ext}	Unitary cost of non-regular capacity

Tableau 0.2 Nomenclature : variables

Variable	Description
$t_i^0, t_i^{C_i+1}$	Starting time and ending time of i
t_i^c	For $c \in 1..C_i$: ending time of part c of i
z_{ip}^c	For $c \in 0..C_i + 1$: binary variable that equals 1 if t_i^c is in period p or before
ϵ_{im}	Binary variable that equals 1 if mode m is chosen for i
d_{ip}^c	Duration of part c of i within period p
d_{ijp}	Duration of overlapping between i and j within period p
l_{ip}^c	Workload of part c of i during period p
$l_{ijp}^{pred}, l_{ijp}^{succ}$	Rework on predecessor i (resp. successor j) during p due to overlapping between i and j
$y_{rp}^{int}, y_{rp}^{ext}$	Regular and non-regular required capacity of resource of resource r in period p
makespan	Project makespan
cost	Project cost

4.1 Continuous/Binary variables constraints

Constraints (19), (20), and (21) are straight-forward results of the definition of z_{ip}^c, and generalize constraints (1) to (3).

$$t_i^c \geq D \cdot p \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in P \tag{19}$$

$$t_i^c \leq D \cdot p + H \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in P \tag{20}$$

$$z_{ip+1}^c \geq z_{ip}^c \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in \{1..|P| - 1\} \tag{21}$$

4.2 Durations over periods

Constraints (22) to (26) provide the link between durations per periods d_{ip}^c and variables t_i^c and z_{ip}^c, similarly to equations (4) to (8) of the original mixed-time model.

$$d^c_{ip} \leq D \cdot (z^{c-1}_{ip} - z^c_{ip-1}) \qquad \forall i \in I, c \in 1..C_i + 1, p \in 1..|P| \qquad (22)$$

$$d^c_{ip} \geq D \cdot (z^{c-1}_{ip-1} - z^c_{ip}) \qquad \forall i \in I, c \in 1..C_i + 1, p \in 1..|P| \qquad (23)$$

$$d^c_{ip} \geq t^c_i - D \cdot p + D \cdot z^{c-1}_{ip-1} - H \cdot (1 - z^c_{ip}) \qquad \forall i \in I, c \in 1..C_i + 1, p \in 1..|P| \qquad (24)$$

$$d^c_{ip} \geq D \cdot p \cdot (1 - z^{c-1}_{ip-1}) - t^{c-1}_i - D \cdot z^c_{ip} \qquad \forall i \in I, c \in 1..C_i + 1, p \in 1..|P| \qquad (25)$$

$$\sum_{p=1}^{|P|} d^c_{ip} = t^c_i - t^{c-1}_i \qquad \forall i \in I, c \in \{1..C_i + 1\} \qquad (26)$$

Constraints (27) to (29) concern overlapping durations within periods by giving upper-bounds for d_{ijp}. Constraints (27) and (28) make sure that overlapping durations within periods never exceed durations within periods of both predecessor i and successor j. Constraints (29) ensure that overlapping durations do not surpass the span between the starting time of j and ending time of i. Note that inequalities (27) consider the assumption that for a given WP, its overlapping predecessors can never overlap on its overlapping successors (no cascade effect).

$$d_{ijp} \leq d^1_{jp} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (27)$$

$$d_{ijp} \leq \sum_{c=1}^{C_i+1} d^c_{ip} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (28)$$

$$d_{ijp} \leq t^{C_i+1}_i - t^0_j + H \cdot (1 - e_{im}) \qquad \forall i \in I, j \in Succ1_i, m \in \{M_i | ch^j_{im} = 1\}, p \in P \qquad (29)$$

4.3 Scheduling constraints

Constraints (30) and (32) force WP i to be in its allowed time window, while constraints (31) make sure that the different parts of WP i are in order. Moreover, the end of each part gives the time t^c_i when a successor in $Succ1_i$ can start. Constraints (33) ensure that successor j begins after the correct milestone of i. For successors in $Succ0_i$ that cannot overlap on i, a classical end-to-start constraint is defined in constraints (34). For successors in $Succ1_i$ that can overlap on i, constraints (35) exclude the case where a predecessor of i and a successor of i overlap (no cascade effect).

$$t_i^0 \geq RD_i \qquad \forall i \in I \tag{30}$$

$$t_i^{c+1} \geq t_i^c \qquad \forall i \in I, c \in \{0..C_i\} \tag{31}$$

$$t_i^{C_i+1} \leq DD_i \qquad \forall i \in I \tag{32}$$

$$t_j^0 \geq t_i^c - H \cdot (1 - c_{im}) \qquad \forall i \in I, m \in M_i, j \in Succ1_i, c \in \{1..C_i + 1 | c = Pos_{ijm}\} \tag{33}$$

$$t_j^0 \geq t_i^{C_i+1} \qquad \forall i \in I, j \in Succ0_i \tag{34}$$

$$t_j^1 \geq t_i^{C_i+1} \qquad \forall i \in I, j \in Succ1_i \tag{35}$$

Figure 0.3 presents an example of a WP (A) with two successors (B, C) that can overlap. A is therefore divided into three parts. In the depicted mode, the milestone corresponding to C comes before the one corresponding to B. Consequently C can start after the end of the first part of A, while B can start after the end of the second part of A.

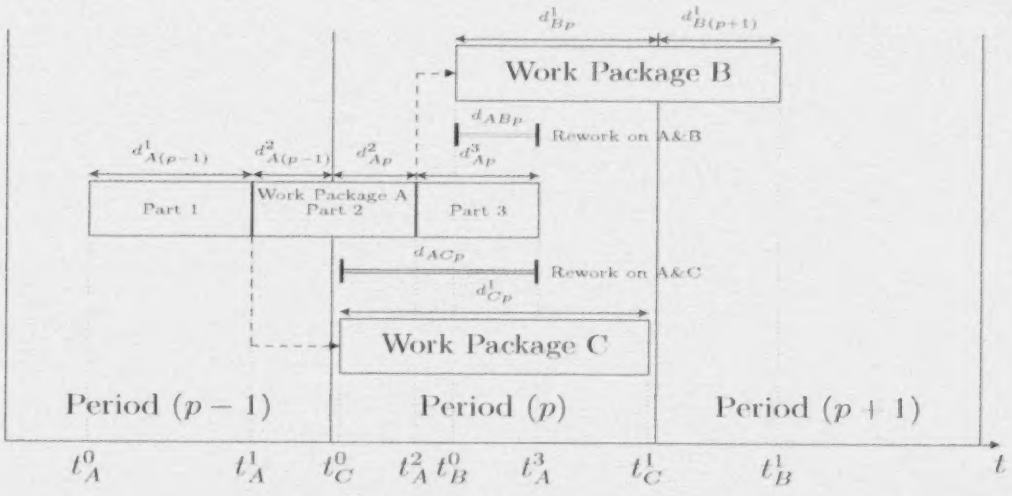

Figure 0.3 Work package (A) with two successors (B, C) that can overlap

4.4 Workload and intensity constraints

Constraints (36) and (37) ensure that the required workload is attained for each part of WP i according to the selected mode.

$$\sum_{p \in P} l_{ip}^{c} \geq L_{im}^{c} \cdot e_{im} \qquad \forall i \in I, c \in \{1..C_i + 1\}, m \in M_i \qquad (36)$$

$$\sum_{p \in P, c \in \{1..C_i+1\}} l_{ip}^{c} = L_i^{total} \qquad \forall i \in I \qquad (37)$$

In addition to initial workloads, reworks are required on predecessors and successors that overlap. The reworks should be executed during the overlapping time between predecessors and successors. Constraints (38) and (39) make sure that the total executed reworks match the required reworks in selected modes.

$$\sum_{p \in P} l_{ijp}^{pred} \geq L_{ijm}^{pred} \cdot e_{im} \qquad \forall i \in I, m \in M_i, j \in Succl_i \qquad (38)$$

$$\sum_{p \in P} l_{ijp}^{succ} \geq L_{ijm}^{succ} \cdot e_{im} \qquad \forall i \in I, m \in M_i, j \in Succl_i \qquad (39)$$

Workload variables $l_{ip}^{c}, l_{ijp}^{pred}$, and l_{ijp}^{succ} are linked to durations d_{ijp} and d_{ip}^{c} to ensure the respect of minimum and maximum allowed intensities. Constraints (40) and (41) guarantee that total assigned workload for WP i in period p stays in the allowed workload window for the total duration d_{ip}^{c} in period p.

$$\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succl_i} l_{ijp}^{pred} + \sum_{i_0 \in Predl_i} l_{i_0ip}^{succ} \leq BJ_i^{max} \cdot \frac{\sum_{c=1}^{C_i+1} d_{ip}^{c}}{D} \qquad \forall i \in I, p \in P \qquad (40)$$

$$\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succl_i} l_{ijp}^{pred} + \sum_{i_0 \in Predl_i} l_{i_0ip}^{succ} \geq BJ_i^{min} \cdot \frac{\sum_{c=1}^{C_i+1} d_{ip}^{c}}{D} \qquad \forall i \in I, p \in P \qquad (41)$$

Constraints (42) and (43) make sure that maximum and minimum intensities are respected for initial workload for every part of a WP, while constraints (44) and (45) ensure that maximum intensity is respected for reworks.

$$l_{ip}^{c} \leq BJ_i^{max} \cdot \frac{d_{ip}^{c}}{D} \qquad \forall i \in I, c \in \{1..C_i+1\}, p \in P \qquad (42)$$

$$l_{ip}^{c} \geq BJ_i^{min} \cdot \frac{d_{ip}^{c}}{D} \qquad \forall i \in I, c \in \{1..C_i+1\}, p \in P \qquad (43)$$

$$l_{ijp}^{pred} \leq BJ_i^{max} \cdot \frac{d_{ijp}}{D} \qquad \forall i \in I, j \in Succl_i, p \in P \qquad (44)$$

$$l_{ijp}^{succ} \leq BJ_j^{max} \cdot \frac{d_{ijp}}{D} \qquad \forall i \in I, j \in Succl_i, p \in P \qquad (45)$$

4.5 Definition-related constraints

The following constraints are straight-forward results of the definition of variables e_{im}, y_{rp}^{int}, y_{rp}^{ext}, *makespan*, and *cost*.

$$\sum_{m \in M_i} e_{im} = 1 \qquad \forall i \in I \qquad (46)$$

$$makespan \geq l_i^{C_i+1} \qquad \forall i \in I \qquad (47)$$

$$y_{rp}^{int} + y_{rp}^{ext} \geq \sum_{i \in I} A_{ir} \cdot \Big(\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succl_i} l_{ijp}^{pred} + \sum_{i_0 \in Predl_i} l_{i_0ip}^{succ} \Big) \qquad \forall r \in R, p \in P \qquad (48)$$

$$y_{rp}^{int} \leq K_{rp}^{int} \qquad \forall r \in R, p \in P \qquad (49)$$

$$cost = C_{ext} \cdot \sum_{r \in R, p \in P} y_{rp}^{ext} \qquad (50)$$

Possible objective functions are project makespan, project cost, or a trade-off between makespan and cost.

5 Instances generation

5.1 Original test instances

In order to test our model, we used a set of 450 instances that were generated by De Boer (1998) and that are commonly used to test RCCP models. Each instance consists in one project and was generated randomly applying a procedure developed by Kolisch et al. (1995).

Using three parameters, De Boer (1998) generated 45 classes of 10 instances each. A class is characterized by the number of WP n, the total number of resources r, and its average slack s. The latter parameter is defined as follow : $s = \frac{\sum_{j \in I} DD_j - RD_j - Dmin_j + 1}{n}$. Three different

values are possible for n (10, 20, or 50 activites), and for r (3, 10, or 20 resources), while parameter s has five possible values (2, 5, 10, 15 and 20).

The following section explains how we modified the instances of De Boer (1998) so as to allow overlapping.

5.2 Modified test instances

We created a program that modifies the original instances of De Boer (1998) in order to handle overlapping. In addition to the previous parameters n, r, and s, we introduced six more parameters for our modified instances generation. These parameters are defined in Table 0.3.

If $p_{\%} = 0$, $Succ0_i$ contains all the successors of WP i and $Succ1_i$ is empty for every WP i. This will result in having instances that are equivalent to the original instances. On the other hand, if $p_{\%} = 1$, $Succ1_i$ contains all the successors of WP i and $Succ0_i$ is empty for every WP i.

If $0 < p_{\%} < 1$, then we randomly choose $[p_{\%} \cdot Total\ Number\ Of\ Couples]$ of the couples to overlap. If the successor is in $Succ1_i$, the couple (i, j) will then have a random number of overlapping modes between Nb_{min} and Nb_{max}.

If a couple $(i, j)|j \in Succ1_i$ has Nb overlapping modes, then the possible overlapping modes would be to begin the successor j after :

$$(100 - n \cdot Ov) \cdot L_i^{total} \qquad \forall n \in 1..Nb \tag{51}$$

Finally, for each mode, the needed rework is calculated as follow :

$$NeededRwk_i = (n \cdot Ov) \cdot \alpha_{rwk}^{pred} \cdot L_i^{total} \qquad \forall n \in 1..Nb \tag{52}$$

$$NeededRwk_j = (n \cdot Ov) \cdot \alpha_{rwk}^{succ} \cdot L_j^{total} \qquad \forall n \in 1..Nb \tag{53}$$

For the reminder of this papaer, we fixed $Nb_{min} = 2$, $Nb_{max} = 3$, $Ov = 20$, and $\alpha_{rwk}^{pred} = \alpha_{rwk}^{succ} = 0.4$ for all generated instances, and only varied parameter $p_{\%}$.

In addition to adding data related to overlapping, we recalculated the release and due dates of the WPs. In fact, time windows were tightened in the original instances, considering simple Finish-to-Start precedence relations De Boer (1998). This makes them inadequate for our instances that allow overlapping. We explain in details our calculations of the release and

Tableau 0.3 Six new parameters for the modified instances generation

Parameter	Description
$p_{\%}$	Percentage of predecessor-successor couples that can overlap. $0 \leq p_{\%} \leq 1$.
Nb_{min}	Minimum number of overlapping modes for every predecessor-successor couple that can overlap. $Nb_{min} \geq 1$.
Nb_{max}	Maximum number of overlapping modes for every predecessor-successor couple that can overlap. $Nb_{max} \leq \frac{100}{Ov}$.
Ov	Percentage that characterizes the amount of overlapping. $0 < Ov \leq 100$.
α_{rwk}^{pred}	Coefficient that characterizes the quantity of needed rework on predecessors. $0 \leq \alpha_{rwk}^{pred} \leq 1$.
α_{rwk}^{succ}	Coefficient that characterizes the quantity of needed rework on successors. $0 \leq \alpha_{rwk}^{succ} \leq 1$.

due dates for our modified instances in section 5.2.

Calculations of release and due dates

In the instances of De Boer (1998) time windows were first calculated using longest path calculations. They were then tightened in order to respect a maximum slack and an average slack. After each modification, all release and due dates were updated using longest path calculations.

For our modified instances, we first spotted all ready and due dates that were not obtained via longest path calculations in the original instances, and we fixed them in our modified instances. We then tightened all time windows using longest path calculations adapted to overlapping. These calculations give for a WP the earliest start date (respectively latest finish date) knowing the earliest start dates (respectively latest finish dates) of all its predecessors (respectively successors) and the maximum possible overlapping with each predecessor (respectively successor). With our adapted calculations of release and due dates, we obtain the same time windows as the original instances when fixing $p_{\%} = 0$ (no overlapping), and possibly larger time windows when $p_{\%} > 0$.

Note that modifying time windows increases the slack values of the instances of De Boer (1998). We calculated the new slack values for the highest $p_{\%}$ value of our new instances (worst case scenario), and found out that the difference between the new and the original values, is equal to 2.35 on average, with a 99% confidence interval of [2.18; 2.52]. This means that even in the worst case, our new instances can still be grouped according to the slack value of their original instances. For the reminder of this paper, and for clarity's sake, we will

keep using the original slack values as parameters for our modified instances.

6 Illustrating example

In this section, only two simple instances (derived from rccp192) are presented for illustration purpose. For each instance, the project consists of 10 WPs, 3 resources, and a time horizon of 23 weeks. The first instance was generated while fixing $p_{\%} = 0$, and the second one was created with $p_{\%} = 0.2$. Figure 0.4 shows the precedence relations between WPs for the two instances. Each vertice represents a WP, and each directed edge represents a predecessor-successor relationship between two WPs. When a predecessor-successor couple has two or more overlapping modes, a label appears on the edge showing the number of overlapping modes between the two WPs.

In the case where $p_{\%} = 0$, no overlapping is allowed between any couple. Thus, all the 12 predecessor-successor couples have one overlapping mode (no overlapping). When $p_{\%} = 0.2$, the instance generator chooses $\lceil 0.2 \cdot 12 \rceil = 3$ couples and randomly gives 2 or 3 overlapping modes for each chosen couple. As it appears in Figure 0.4, couples WPs 1-4 and WPs 6-10 have 2 overlapping modes each : WPs 4 and 10 can both start after 80%, or 100% of WPs 1 and 6 (respectively) are completed. Couple WPs 2-5 have 3 overlapping modes : WP 5 can start after 60%, 80%, or 100% of WP 2 is attained. Thus WPs 1 and 6 have 2 overlapping modes with their successors, while WP 2 has 3 overlapping modes. Furthermore, each of WPs 1, 2 and 6 is divided into 2 parts, as they only have one successor that can overlap.

We implemented our model using the Optimization Programming Language (OPL) and ran our model on IBM ILOG CPLEX Optimization Studio 12.5.1.0 while minimizing a tradeoff between project cost and makespan. We also implemented a code in order to visualize the solution with a Gantt chart.

Figure 0.5 shows the charts that we obtained with both instances, as well as the allocations for the three resources. WPs are shown in descending order in the Gantt chart (WP 1 is the highest, and WP 10 is the lowest). For each WP, a white rectangle delimits the allowed time window. Note that these rectangles are wider for $p_{\%} = 0.2$ because ready and due dates were recalculated in order to take into account the possibility of overlapping. Colored rectangles show starting and ending dates of WPs (or possibly of parts of WPs). Resource allocations are shown per period with stacked bar charts, where each color relate to a WP in the Gantt chart, and where hatched bars refer to rework. Stepped curves represent regular resource availability for each period.

Both instances were solved to optimality in less than 4 seconds. When allowing overlapping,

project makespan was reduced at the price of increasing project cost. In Figure 0.5, we see that WP 1 overlaps on WP 4 thus entailing rework workloads for both WPs. This additional workload is allocated during period 6. Note that, because of overlapping, WPs 8 and 9 finished earlier, thus reducing project duration by 2.3 periods (11%). Meanwhile project cost is increased by 43.3 units (34%) because of a greater use of non regular resources.

However, minimizing tradeoff function does not always have the same effect on project cost and makespan as in this example. Other instances showed that overlapping can reduce the use of non regular resources, thus diminishing project cost, at the expense of an increased project duration.

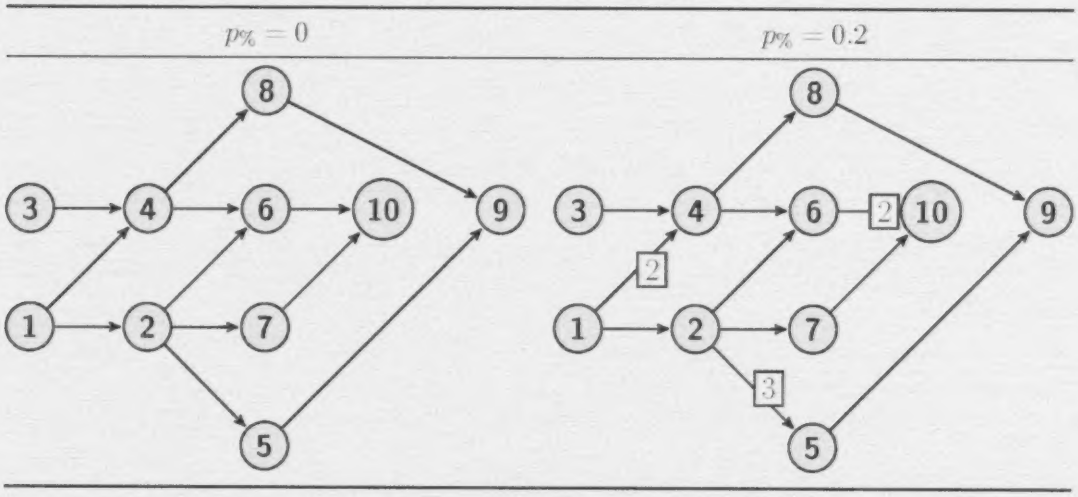

Figure 0.4 Precendence relations between WPs for a project with and without overlapping

7 Computational results

We performed all our tests on a single thread, using a computational grid consisting of 26 PCs with two 3.07 GHz Intel(R) Xeon(R) X5675 Processors under Linux. We encoded our model using the Optimization Programming Language (OPL) and ran our model on IBM ILOG CPLEX Optimization Studio 12.5.1.0 using the default values for all CPLEX parameters. We first ran our model on five sets, with five different values of $p_{\%}$ (0, 0.1, 0.2, 0.3 and 0.4), of 450 instances each (total of 2250 instances), with a time limit of 10000 seconds for every test. We then divided the time limit into 20 intervals of 500 seconds each, and counted the number of times CPU time was inside each interval. Figure 0.6 shows that 56.9% of instances were solved to optimality during the first 500 seconds, and that time limit was reached before optimality for 32.2% of instances. Figure 0.6 also shows that only 2% of instances

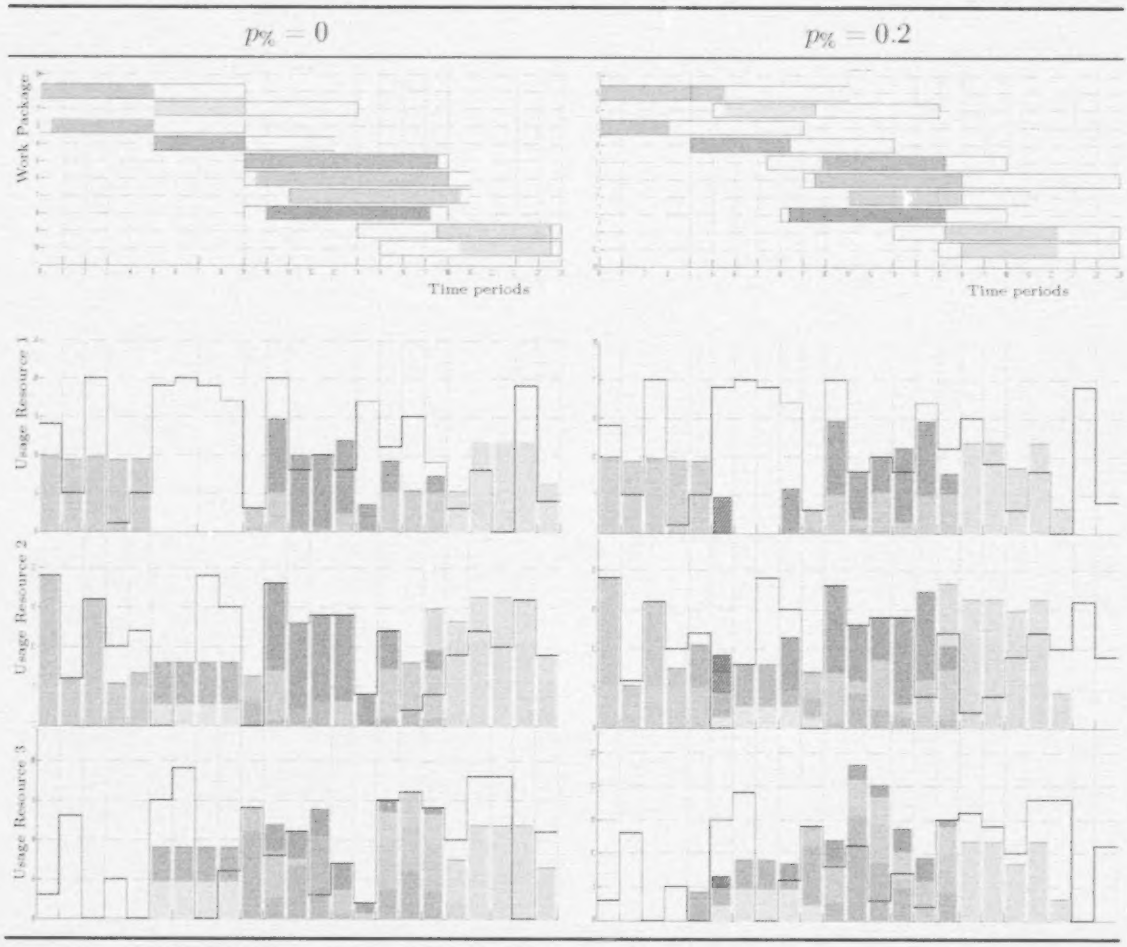

Figure 0.5 Gantt chart, with usage per period of the three resources

were solved to optimality between 5000 and 10000 seconds. Consequently, in the remaining tests, we terminated the search after 5000 seconds for every test.

7.1 Performance analysis

A series of tests was conducted in order to evaluate the performance of our new model. We denote by A the original model of Haït and Baydoun (2012) and B our new model that handles overlapping. We ran the model A on the 450 instances of De Boer (1998), and our model B on our modified instances of De Boer (1998) where we fixed $p_{\%} = 0$. In this particular case where no overlapping couples are allowed, the two sets of instances are equivalent. Having the same optimal solutions, we only compare the performance of the two models in terms of CPU times, the gaps between lower and upper bounds, and the number of times the models proved

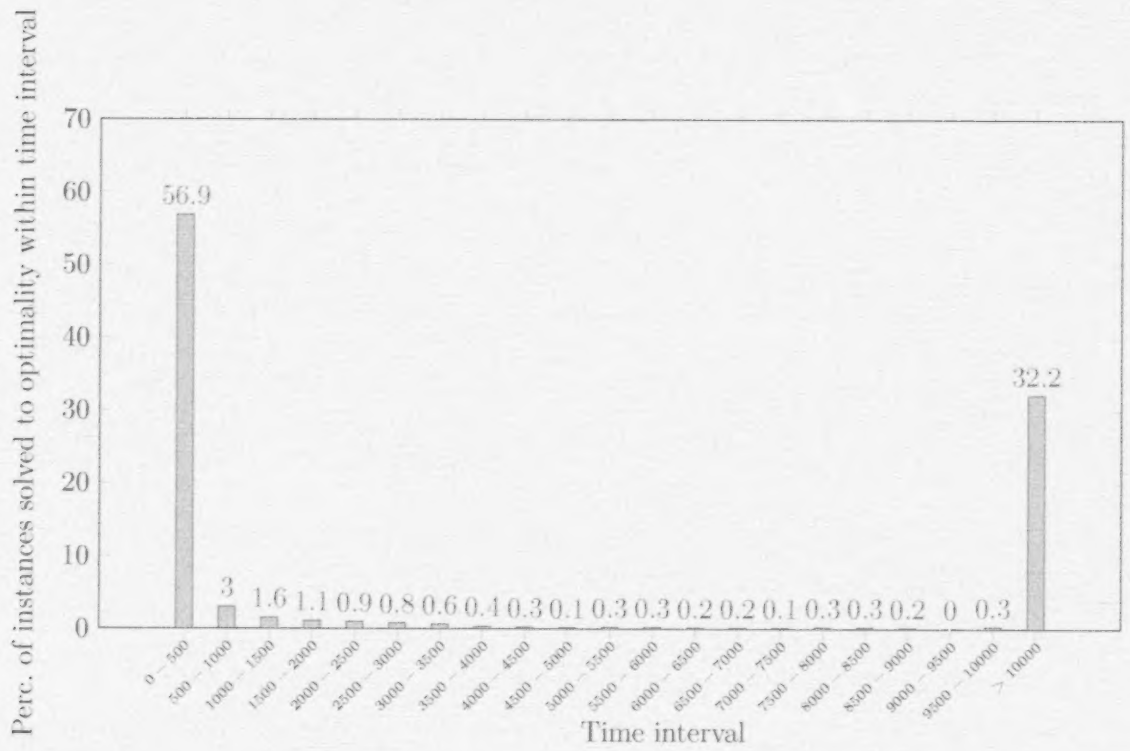

Figure 0.6 Percentage of instances that were solved to optimality within the specified time interval

optimality. Table 0.4 shows for each class the average CPU time for the original model A and our new model B, with the objective of minimizing project cost, while Table 0.5 compares the number of times both models A and B found an optimal solution for each class. Tables 0.4 and 0.5 show that the performance of new model B is not degraded compared to model A when tested on equivalent instances.

We also tested our new model B on five sets of instances with different values of $p_{\%}$, when minimizing project cost. We then defined five criteria for evaluating the performance of the model for each value of $p_{\%}$: the percentage of instances that were not solved to optimality, the average CPU time divided by time limit, the percentage of instances that needed more than 50% of the time limit in order to be solved to optimality, the average gap, and the percentage of instances that have more than 5% of gap when reaching time limit. Figure 0.7 presents the results for each value of $p_{\%}$ in a radar chart, where better performing set of instances are closer to the center. Figure 0.7 shows that the performance of our model is degraded on average, according to the five defined criteria, when the percentage of predecessor-successor

Tableau 0.4 Average CPU time in seconds for models A and B when minimizing the project cost without overlapping

	$n = 10$			$n = 20$			$n = 50$		
$r =$	3	10	20	3	10	20	3	10	20
$s = 2$									
A	0.1	0.1	0.1	0.1	0.2	0.2	0.4	0.6	1
B	0.1	0.1	0.1	0.2	0.2	0.2	0.7	0.9	1.1
$s = 5$									
A	0.3	0.4	0.4	1	2.1	3.8	14	289	280
B	0.3	0.4	0.5	1.6	2.5	4	30	406	346
$s = 10$									
A	3.5	2.5	3	39	223	235	2694	4930	5000
B	4.9	2.7	3.1	56	383	187	3488	5000	5000
$s = 15$									
A	6	10	22	450	2147	3115	5000	5000	5000
B	11	12	21	636	2217	2980	5000	5000	5000
$s = 20$									
A	13	66	48	1036	4198	3818	5000	5000	5000
B	22	74	48	2071	4315	3771	5000	5000	5000

Tableau 0.5 Number of times models A and B proved optimality when minimizing the project cost without overlapping

	$n = 10$			$n = 20$			$n = 50$		
$r =$	3	10	20	3	10	20	3	10	20
$s = 2$									
A	10	10	10	10	10	10	10	10	10
B	10	10	10	10	10	10	10	10	10
$s = 5$									
A	10	10	10	10	10	10	10	10	10
B	10	10	10	10	10	10	10	10	10
$s = 10$									
A	10	10	10	10	10	10	6	1	0
B	10	10	10	10	10	10	4	0	0
$s = 15$									
A	10	10	10	10	7	7	0	0	0
B	10	10	10	10	8	6	0	0	0
$s = 20$									
A	10	10	10	10	4	3	0	0	0
B	10	10	10	8	2	4	0	0	0

couples that can overlap augments. This is the result of an increase in the number of binary and continuous variables, due to the separation of some WPs into several parts, and also due to the different possible overlapping modes.

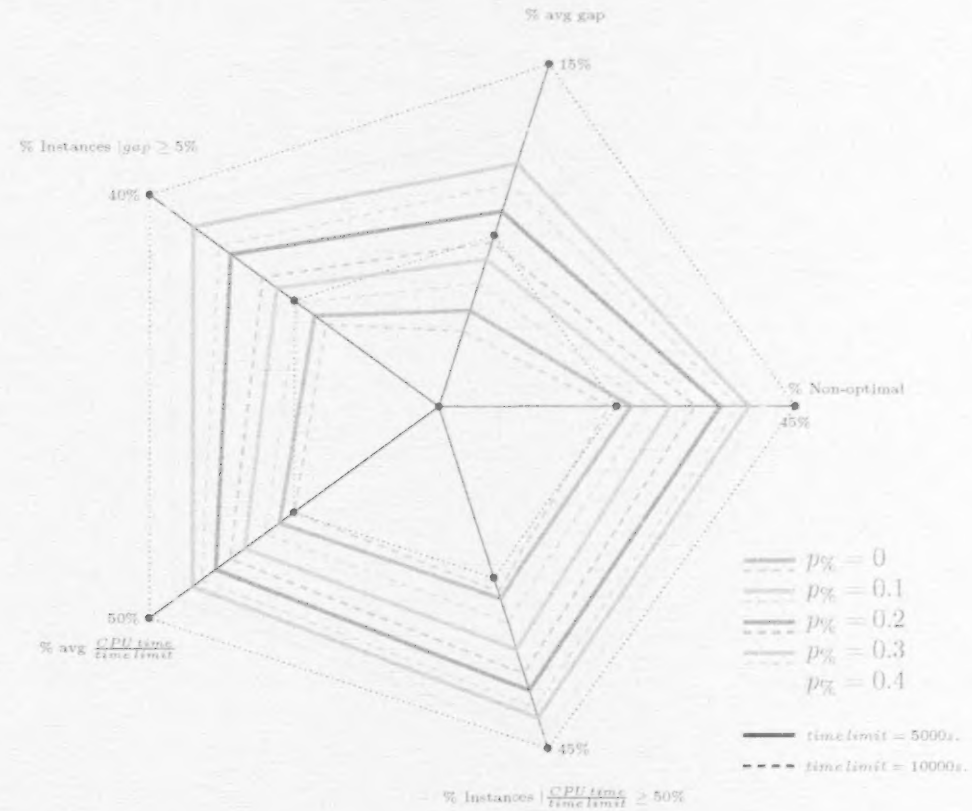

Figure 0.7 Impact of changing $p_{\%}$ on the performance of B, when minimizing project cost

It is clear that the new parameter $p_{\%}$ is not the only factor that has an impact on CPU time; all the four aforementioned parameters of instances can affect the performance of the model. Figure 0.8 shows for each parameter s, r, n and $p_{\%}$, the average CPU time, with a 95% confidence interval. It suggests that parameters n and s have the biggest impact on the performance, followed by $p_{\%}$, and lastly r.

We first studied the effects of parameters s, r, n and $p_{\%}$, on having a CPU time bigger than time limit. For that matter, we created a binary variable that separates cases where CPU time is bigger than time limit (denoted event E_1), from those where CPU time is smaller (denoted event E_2). We then performed a Logistic regression on this binary variable, with parameters s, r, n and $p_{\%}$ as continuous predictors. The Logistic regression correctly predicts 94% of events E_2, and 82% of events E_1. The area under ROC curve is 0.964 (Figure 0.9),

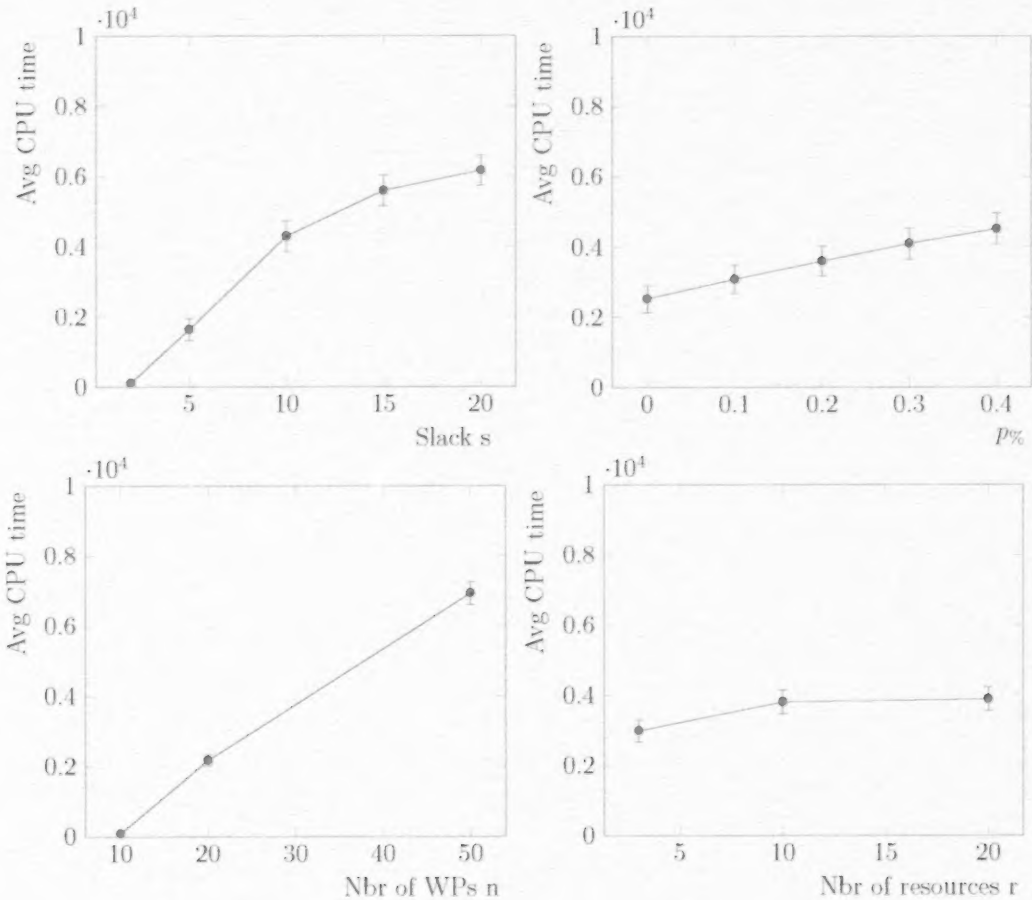

Figure 0.8 Average CPU time, with 95% confidence level for different values of four instance parameters

meaning that the regression has good accuracy. Odds ratios in Table 0.6 show the relative ratios of odds of event E_1 when increasing the predictor of one unit. All four values are bigger than 1, meaning that increasing any parameter augments the probability of event E_1. Also note that a unitary increase of s or n has significantly more impact on odds of event E_1 than a unitary increase of parameters r or $p_{\%}$.

We also conducted an ANOVA study on 1522 instances where CPU time did not reach time limit. This study gives results that are in line with our assumptions. In fact, Pareto chart of standardized effects (Figure 0.10) shows that factor s has the most significant impact on CPU time, followed by n, $p_{\%}$, and lastly r.

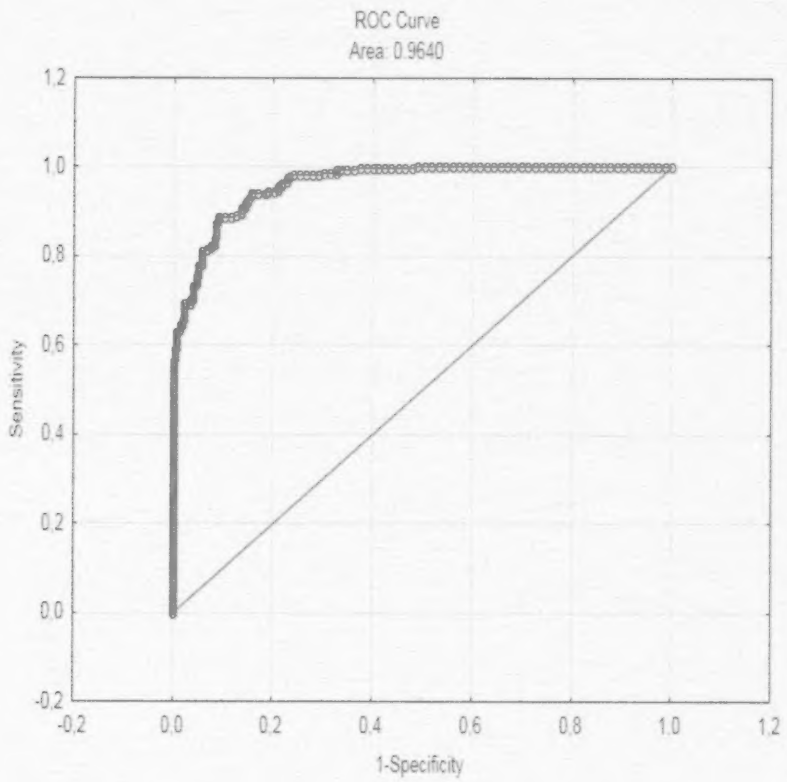

Figure 0.9 Receiver Operating Characteristic (ROC) curve for the Logistic regression

7.2 Results analysis

In order to conduct an analysis on the results that were obtained, we ran our model on the 450 instances with $p_{\%} = 0$ (equivalent to the original instances of De Boer (1998)) two times. We first fixed an objective of minimizing project cost, and then we ran the model another time while minimizing project delivery time. Among 450, only 340 were solved to optimality when minimizing project cost. For these instances, we obtained the optimal cost $Cost_0$ and makespan $Makespan_0$. We integrated these two parameters in all the corresponding data files (with the five different values of $p_{\%}$). This means that for a given instance, we added two additional parameters, $Cost_0$ and $Makespan_0$, which are the minimum possible cost and makespan for the instance if no overlapping was allowed.

We used these two parameters in order to create a tradeoff objective function (54) with normalized project cost and delivery time.

Tableau 0.6 Odds ratios of event E_1 for a unitary increase in parameters

Parameter	Odds ratio
s	1.645707
n	1.226389
r	1.079625
$p\%$	1.075126

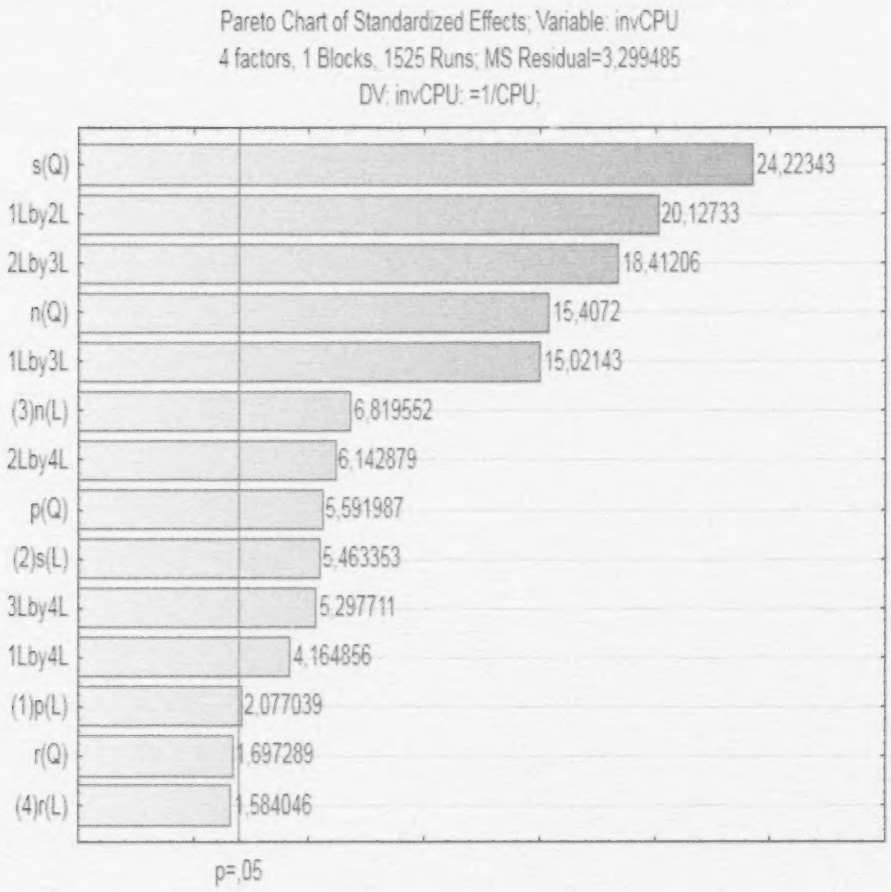

Figure 0.10 Pareto chart of standardized effects for factors n, s, $p\%$, and r, and dependent variable $log(1/CPU\ time)$

$$minimize \quad \beta_{cost} \cdot \frac{cost}{Cost_0} + \beta_{makespan} \cdot \frac{makespan}{Makespan_0} \tag{54}$$

For each instance we tested three combinations for the coefficients β_{cost} and $\beta_{makespan}$ as

depicted in Table 0.7. For each combination, and each non-zero value of $p_{\%}$, we compared the results to the case where no overlapping was allowed ($p_{\%} = 0$). Figure 0.11 shows for each combination and non-zero value of $p_{\%}$, the percentage of instances where the objective was improved compared to the case where no overlapping was allowed.

Two remarks can be made concerning the shape of the bar graph 0.11. First, note that, for each combination of values for β_{cost} and $\beta_{makespan}$, the percentage of instances where the objective was improved when allowing overlapping, increases with $p_{\%}$. This can be explained by the fact that the bigger $p_{\%}$ is, the more flexible the problem becomes. Thus, it is more likely to have a smaller objective value with bigger $p_{\%}$. Secondly, note that for a given $p_{\%}$, the percentage of instances that have better objective with overlapping is bigger when $\beta_{cost} = 25$ and $\beta_{makespan} = 75$ (combination 1) than the case where $\beta_{cost} = 75$ and $\beta_{makespan} = 25$ (combination 3). This can be explained by the fact that overlapping is mainly beneficial when accelerating project delivery time. Thus its favorable effects are more pronounced when the tradeoff puts more emphasis on makespan than on cost. However, even though overlapping entails additional workload, it is still beneficial for having smaller project cost, by adding a flexibility in distributing the workload during regular resource capacities.

8 Conclusion

Motivated by the common use of concurrent engineering methods in construction projects, we proposed an interesting extension of the RCCP that allows overlapping of WPs. Five sets of 450 modified RCCP instances were created and let us conduct two types of analysis on our model. The performance analysis showed that our model is concurrent with the original model when our modified instances are equivalent to the original ones, and becomes less efficient when the percentage of possible overlapping couples is increased. However, this slower performance is the price to pay in order to have smaller project delivery time and cost, as it was shown by our results analysis. In fact, our results showed that overlapping is beneficial for accelerating project delivery times, and also for reducing project cost by allowing a better distribution of workload.

Tableau 0.7 Three combinations for coefficients β_{cost} and $\beta_{makespan}$

	β_{cost}	$\beta_{makespan}$
Combination 1	25	75
Combination 2	50	50
Combination 3	75	25

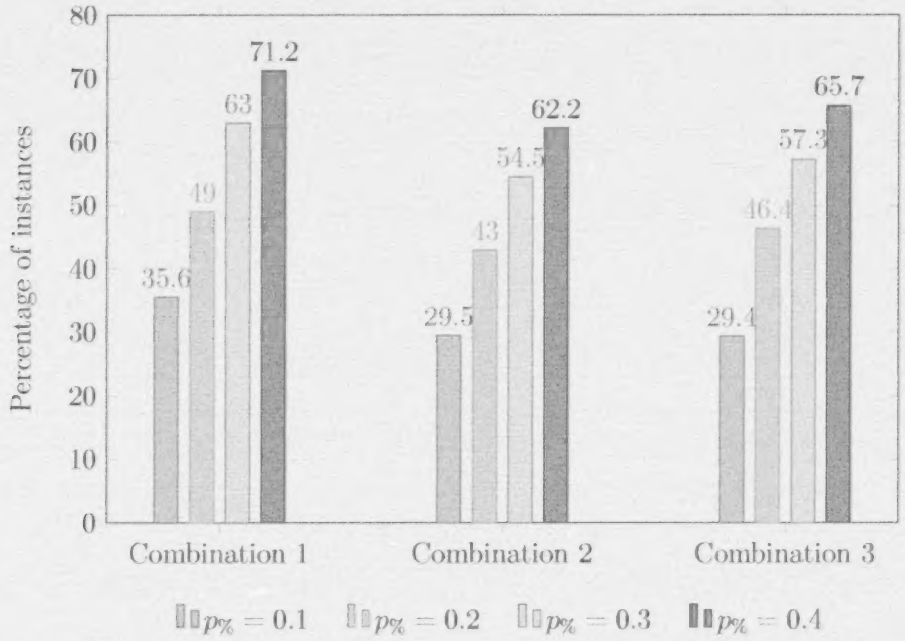

Figure 0.11 Percentage of instances where overlapping improved the objective

Future work could focus on ameliorating the performance of the model, by exploring alternative modeling and solving techniques. Another avenue for research could be the development of a model-based heuristic dedicated for the hard instances, in order to give acceptable solutions in reasonable time.

acknowledgements

This work has been supported by the Natural Sciences and Engineering Research Council of Canada and the Jarislowsky/SNC-Lavalin Research Chair in the Management of International Projects. Their support is gratefully acknowledged.

RÉFÉRENCES

A. Alfieri, T. Tolio, et M. Urgo, "A project scheduling approach to production planning with feeding precedence relations," *International Journal of Production Research*, vol. 49, no. 4, pp. 995–1020, 2011.

F. Berthaut, P. Robert, P. Nathalie, et H. Adnène, "Time-cost trade-offs in resource-constraint project scheduling problems with overlapping modes," *International Journal of Project Organisation and Management*, vol. 6, no. 3, pp. 215–236, 2014.

L. Bianco et M. Caramia, "A new formulation for the project scheduling problem under limited resources," *Flexible Services and Manufacturing Journal*, vol. 25, no. 1-2, pp. 6–24, 2013. [En ligne]. Disponible : http://dx.doi.org/10.1007/s10696-011-9127-y

K. Cherkaoui, R. Pellerin, P. Baptiste, et N. Perrier, "Planification hiérarchique de projets EPCM," dans *10ème Conférence Internationale de Génie Industriel 2013*, La Rochelle, France, 12–14 Jun. 2013.

R. De Boer, "Resource-constrained multi-project management : a hierarchical decision support system," Thèse de doctorat, University of Twente, Enschede, The Netherlands, Oct. 1998.

N. Gademann et M. Schutten, "Linear-programming-based heuristics for project capacity planning," *IIE Transactions*, vol. 37, no. 2, pp. 153–165, 2005.

J. E. V. Gerk et R. Y. Qassim, "Project Acceleration via Activity Crashing, Overlapping, and Substitution," *IEEE Transactions on engineering management*, vol. 55, no. 4, pp. 590–601, Nov. 2008.

L. Grèze, R. Pellerin, P. Leclaire, et N. Perrier, "A heuristic method for resource-constraint project scheduling with activity overlapping," *Journal of Intelligent Manufacturing*, 2012, forthcoming.

E. Hans, "Resource loading by Branch-and-Price techniques," Thèse de doctorat, University of Twente, Enschede, The Netherlands, Oct. 2001.

A. Haït et G. Baydoun, "A new event-based MILP model for the resource-constrained project scheduling problem with variable intensity activities," dans *The IEEE International Conference on Industrial Engineering and Engineering Management 2012*, Honk Kong, 10–13 Déc. 2012.

T. Kis, "A branch-and-cut algorithm for scheduling of projects with variable-intensity activities," *Mathematical programming*, vol. 103, no. 3, pp. 515–539, 2005.

T. Kis, "RCPS with variable intensity activities and feeding precedence constraints," dans *Perspectives in modern project scheduling*, J. Józefowska et J. Weglarz, éds. Berlin : Springer, 2006, ch. 5, pp. 105–129.

R. Kolisch, A. Sprecher, et A. Drexl, "Characterization and generation of a general class of resource-constrained project scheduling problems," *Management science*, vol. 41, no. 10, pp. 1693–1703, 1995.

A. Naber et R. Kolisch, "MIP models for resource-constrained project scheduling with flexible resource profiles," *European Journal of Operational Research*, vol. 239, no. 2, pp. 335–348, 2014. [En ligne]. Disponible : http://dx.doi.org/10.1016/j.ejor.2014.05.036

A. Pritsker et L. Watters, *A Zero-one Programming Approach to Scheduling with Limited Resources*, sér. Memorandum (Rand Corporation) RM-5561-PR. Rand Corporation, 1968.

G. Wullink, "Resource loading under uncertainty," Thèse de doctorat, University of Twente, Enschede, The Netherlands, Mar. 2005.